Luigi Buglione

Some thoughts on Productivity in ICT projects

White Paper

Version 1.2 - July 25, 2008

How to reference this document:
Luigi Buglione, *Some thoughts on Productivity in ICT Projects, version 1.2,* WP-2008-02, White Paper, July 25 2008

For more information about PSU and other Software Measurement & Quality issues, please visit:
< http://www.geocities.com/lbu_measure > or contact the Author by email at luigi.Buglione@computer.org

First Printing: July 2008

Table of Contents

1 Document Information

1.1 Executive Summary

This document proposes a discussion about the current, common definitions of productivity within the Software-Systems Engineering community, stressing some practical incongruities across some ISO/IEC standards and proposing some ways to overcome them.

1.2 History

Revision	Date	Changes since last revision
1.00	July 1, 2007	• First issue
1.10	March 4, 2008	• Main changes: general improvement of existing contents, introduced the definition for "Project Size", split Section 2 into three sub-sections, added Sections 3.4, corrected some typo errors, acronyms and references added in the list, added Section 3.5 and 3.6; repositioned section 6.3 as Section 7.
1.20	July 25, 2008	• Section 2.2: update the 'project size' definition and inserted the SEVOCAB reference; added a project-product comparison example • Section 3.5: added sources for different sizing approach for NFR • Section 3.6: added references

1.3 Acronyms

Acronym	Description
AKA	Also Known As
CMMI	Capability Maturity Model Integration (www.sei.cmu.edu/cmmi/)
COCOMO	Cost Construction Model (http://sunset.usc.edu/research/COCOMOII)
COSMIC	Common Software International Consortium (www.cosmicon.com)
CSMS	Certified Software Measurement Specialist
F/Q/T	Functional / Quality / Technical (referred to the nature of a requirement)
F/Q/T/O	Functional / Quality / Technical / Other (referred to the nature of a requirement)
FFP	Full Function Points
FP	Function Points
FPA	Function Point Analysis
FSM	Functional Size Measurement
FSMM	FSM Method
FTE	Full Time Equivalent
FUR	Functional User Requirement
GSC	General System Characteristic
GUFPI-ISMA	Gruppo Utenti Function Point Italia – Italian Software Metrics Association (www.gufpi-isma.org)
HLR	High-Level Requirement
ICT	Information & Communication Technology
IFPUG	International Function Point User Group (www.ifpug.org)
IPO	Input-Processing-Ouput
ISBSG	International Standard Benchmarking Software Group (www.isbsg.org)
ISO	International Organization for Standardization (www.iso.org)
JTC	Joint Technical Committee
LOC	Lines of Code
ML	Maturity Level
MRE	Mean Relative Error
NFR	Non-Functional Requirement
NESMA	Netherlands Software Metrics Association (www.nesma.org)
PA	Process Area
PM	Project Manager

PMBOK	Project Management Body of Knowledge (www.pmi.org)
PMC	Project Monitoring & Control
PP	Project Planning
PSU	Project Size Unit (http://www.geocities.com/lbu_measure/psu/psu.htm)
RD	Requirement Development
RHLR	Refined HLR
SC	Sub-Committee
SEVOCAB	Software & Systems Engineering Vocabulary (http://pascal.computer.org/sev_display/index.action)
SLC	Software Life Cycle
SP	Specific Practice
SPICE	Software Process Improvement Capability dEtermination (www.isospice.com)
STAR	Software Taxonomy Revised
TC	Technical Committee
UFP	Unadjusted FP
UKSMA	United Kingdom Software Metrics Association (www.uksma.co.uk)
UR	User Requirement
VAF	Value Adjustment Factor
WG	Working Group

1.4 References

[ALBR79]	Albrecht A.J., *Measuring Application Development Productivity*, Proceedings of the IBM Applications Development Symposium, GUIDE/SHARE, October 14-17, 1979, Monterey, CA, pp. 83-92
[ALBR84]	Albrecht A.J., *AD/M Productivity Measurement and Estimate Validation*, IBM Corp., NY, 1984
[BOEH81]	Boehm B., *Software Engineering Economics*, Englewood Cliffs N.J., Prentice-Hall Inc., 1981, ISBN 0138221227
[BOEH00]	Boehm B.W., Horowitz E., Madachy R., Reifer D., Clark B.K., Steece B., Brown A.W., Chulani S & Abts C., *Software Cost Estimation with COCOMOII*, Prentice Hall, 2000, ISBN 0130266922
[BUGL02]	Buglione L. & Abran A., *ICEBERG: a different look at Software Project Management*, IWSM2002 in "Software Measurement and Estimation", Proceedings of the 12th International Workshop on Software Measurement (IWSM2002), October 7-9, 2002, Magdeburg (Germany), Shaker Verlag, ISBN 3-8322-0765-1, pp. 153-167, URL: www.lrgl.uqam.ca/publications/pdf/757.pdf
[BUGL05]	Buglione L. &Abran A., *A Model for Performance Management & Estimation*, Proceedings of METRICS 2005, 11th IEEE International Software Metrics Symposium, 19-22 September 2005, Como(Italy), ISBN 0-7695-2371-4, URL: http://doi.ieeecomputersociety.org/10.1109/METRICS.2005.3
[BUGL07]	Buglione L,, *PSU Measurement Manual, v1.21,* , URL: www.geocities.com/lbu_measure/psu/psu.htm
[BUGL08a]	Buglione L., *Misurare il Software. Quantità, qualità, standard e miglioramento di processo nell'ICT*, 3° Edition, FrancoAngeli, FA724.20, January 2008, ISBN: 978-88-464-9271-5, URL: www.geocities.com/lbu_measure/libri/mis.htm
[BUGL08b]	Buglione L., *Improving Estimation by Effort Type Proportions*, Software Measurement News, Vol. 13, No.1, Spring 2008, URL: http://ivs.cs.uni-magdeburg.de/sw-eng/agruppe/forschung/
[BUSH90]	Bush M.E. & Fenton N.E., *Software Measurement : A Conceptual Framework*, The Journal of Systems and Software, Elsevier Science North-Holland, Vol. 12 No. 3, July 1990, pp. 223-232, URL: http://portal.acm.org/citation.cfm?id=82645&jmp=citings&dl=GUIDE&dl=GUIDE
[COSM07]	COSMIC, *The COSMIC Fuunctional Size Measurement Method. Version 3.0 – Measurement Manual,* September 2007, URL: www.cosmicon.com
[DCG07]	DCG, *Comparative Sizing and Measurement is Critical to the Improvement of Software Application Development and Maintenance*, White Paper, 2007, www.davidconsultinggroup.com/measurement/industry_data.aspx
[DEKK07]	Dekkers C., *FP Chaos – Making Sense of Zero FP Projects*, Proceedings of the SMEF 2007 (4th Software Measurement European Forum), Rome (Italy), 9-11 May 2007, pp. 307-312, URL: www.dpo.it/smef2007/papers/day3/303.pdf
[DEKK08]	Dekkers T., *Sizing to estimate the complete IT project*, Proceedings of SMEF2008 (4th Software Measurement European Forum), Milan (Italy), 28-30 May 2008, URL: www.dpo.it/smef2008/presentazioni/SMEF08_306_Dekkers.pdf
[DERY05]	Déry D. & Abran A., *Investigation of the Effort Data Consistency in the ISBSG Repository*, in 15th International Workshop on Software Measurement - IWSM'2005 , Montreal, Canada, Shaker-Verlag , 2005, pp. 123-136, URL: http://www.lrgl.uqam.ca/publications/pdf/909.pdf
[ECSS05]	ECSS, *Space Engineering – System Engineering: Part 6. Functional and Technical Specifications*, European Cooperation for Space Standardization, ECSS-E-10 Part 6A rev.1, October 31 2005, URL: www.ecss.nl
[EELE05]	Eeles P., *Capturing Architectural Requirements,* IEEE DeveloperWorks, 2005, URL: www-128.ibm.com/developerworks/rational/library/4706.html
[GLIN05]	Glinz M., *Rethinking the Notion of Non-Functional Requirements*, Proceedings of the 3rd World Congress on Software Quality (3WCSQ), Munich (Germany), September 2005, URL: www.ifi.uzh.ch/rerg/fileadmin/downloads/publications/papers/3WCSQ2005.pdf
[GRAD87]	Grady R. & Caswell D., *Software Metrics: Establishing a Company-Wide Program*., Prentice Hall, 1987, ISBN 0138218447.
[IEEE98]	IEEE, STD-1058-1998, *Standard for Software Project Management Plans*, 1998

[IFPU03]	IFPUG, *Framework for Functional Sizing*, Version 1.0, September 2003*),* International Function Point User Group, Westerville, Ohio, January 2004, URL: www.ifpug.org
[IFPU04a]	IFPUG, *Function Points Counting Practices Manual (release 4.2),* International Function Point User Group, Westerville, Ohio, January 2004, URL: www.ifpug.org
[IFPU04b]	IFPUG, *Guidelines to Software Measurement (release 2),* International Function Point User Group, Westerville, Ohio, July 2004, URL: www.ifpug.org
[ISBS06]	ISBSG, *Glossary of terms, version 5.9.1*, January 2006, URL: www.isbsg.org/html/Glossary_of_Terms.doc
[ISBS07]	ISBSG, *ISBSG Repository R10 Field Description*, January 2007, URL: www.isbsg.org/html/R9%20Field%20Descriptions.doc
[ISO95]	ISO/IEC JTC1/SC7/WG7 N72, *International Standard 12207 - Information Technology : Software Life Cycle Processes*, 22/02/95, URL: ww.iso.ch
[ISO98]	ISO/IEC14143-1:1998, *Information Technology - Software Measurement-Functional Size Measurement - Part 1: Definitions of Concepts*: International Organization for Standardization, 1998, URL: www.iso.ch
[ISO01]	ISO/IEC 9126-1:2001, *Software Engineering-Product Quality-Part 1: Quality Model*: ISO and IEC, 2001, URL: www.iso.ch
[ISO02]	ISO/IEC 20968:2002, *Software Engineering-MK II Function Point Analysis- Counting Practices Manual*: ISO and IEC, 2002, URL: www.iso.ch
[ISO03a]	ISO/IEC 20926:2003, *Software engineering -- IFPUG 4.1 Unadjusted functional size measurement method -- Counting practices manual*, 2003, URL: www.iso.ch
[ISO03b]	ISO/IEC 19761:2003, *Software Engineering-Cosmic FFP-A functional Size Measurement Method*: ISO and IEC, 2003, URL: www.iso.ch
[ISO05a]	ISO/IEC, IS 24570:2005 - Software engineering -- NESMA functional size measurement method version 2.1 -- Definitions and counting guidelines for the application of Function Point Analysis, International Organization for Standardization, 2005, URL: www.iso.ch
[ISO05b]	ISO, IS 9000:2005, *Quality Management Systems – Fundamentals & Vocabulary*, Genève, 2005, URL: www.iso.ch
[ISO05c]	ISO, IS 21351:2005, *Space Systems – Functional and Technical Specifications*, May 19, 2005, URL: www.iso.ch
[ISO06]	ISO/IEC, TR 15504-5, *Information technology -- Process Assessment -- Part 5: An exemplar Process Assessment Model,* 2006, URL: www.iso.ch
[ISO07]	ISO/IEC14143-1:2007, *Information Technology - Software Measurement-Functional Size Measurement - Part 1: Definitions of Concepts*: International Organization for Standardization, 2007, URL: www.iso.ch
[JONE97]	JONES C., *What are Function Points?*, Software Productivity Research Inc., 1997, URL: www.spr.com/products/function.htm
[LANZ08]	LANZA G., *Function Point: how to transform them in effort? That's the problem!*, Proceedings of SMEF2008 (4th Software Measurement European Forum), Milan (Italy), 28-30 May 2008, URL: http://www.dpo.it/smef2008/papers/SMEF08_proc_203_LANZA.pdf
[PAUL93]	PAULK M.C., WEBER C.V., GARCIA S.M., CHRISSIS M.B. & BUSH M., *Key Practices of the Capability Maturity Model Version 1.1*, Software Engineering Institute/Carnegie Mellon University, CMU/SEI-93-TR-25, February 1993, URL: www.sei.cmu.edu/pub/documents/93.reports/pdf/tr25.93.pdf
[PMI04]	PMI, *A Guide to the Project Management Body of Knowledge* (PMBOK), 2004 Edition, Project Management Institute, 2004, URL: www.pmi.org
[SEI06]	CMMI PRODUCT DEVELOPMENT TEAM, *CMMI for Development Version 1.2,* CMU/SEI-2006-TR-008, Technical Report, Software Engineering Institute, August 2006, URL: www.sei.cmu.edu/pub/documents/06.reports/pdf/06tr008.pdf
[SEVO08]	ISO/IEEE, *SEVOCAB : Software & Systems Engineering Vocabulary*, 2008, URL: http://pascal.computer.org/sev_display/index.action

2 Introduction

2.1 What is productivity?

Productivity is one of the general-purpose concepts useful for any planning and monitoring activity. Its widely accepted definition (*the amount of output created- in terms of goods produced or services rendered - per unit input used*), as stated in the PMBOK guide and in the Software-Systems Engineering domain is referenced and cited in SPI models - such as ISO/IEC 15504 [ISO06] or CMMI [SEI06] - as one of the main values to take into account for planning and/or monitoring a software project.

From the late '70s, functional size measures such as Albrecht's Function Points have been proposed as a new way to size the functional side of a software in lieu of Lines of Code (LOC) [ALBR79][ALBR84]. Productivity can then be calculated as the ratio between the number of functional size units and the whole project effort (in hours or days) IFPUG [IFPUG04a] and ISBSG [ISBSG07]. The adoption of Functional Size in the Software-Systems Engineering community was the starting point in the early '90s for the creation of an ISO/IEC JTC1/SC7 working group (WG12) with the goal to create a set of common criteria for validating a Functional Size Measurement Method (FSMM), and led to the adoption of FSMM *de jure* standards.

In particular, ISO/IEC 14143-1 [ISO98][ISO07] provides the ISO definition of the Functional Size concepts and provides a basis against which all FSMM variants can be compared; it also provides a process for checking whether a Candidate FSM Method conforms to the provisions of ISO/IEC 14143-1. ISO stated that there are three types of requirement (Functional, Quality, and Technical), and that only Functional Requirements are valid inputs for calculating a FSM unit.

Therefore in the IFPUG CPM v4.2 (ISO 20926:2003) [ISO03a] only its *unadjusted* version (excluding VAF – Value Adjustment Factor – therefore the non-functional attributes and related sizes) is recognized by ISO as a valid FSMM.

Furthermore, a FSMM expresses one of the *product* measures within a software project and it is not a *project* measure by itself; the consequence is that sizes, efforts and productivities within a project can be categorized into functional and non-functional parts.

Again, shaping and delimiting the scope of the productivity definition, it must be reminded that *productivity* and *performance* are two related, but different concepts: performance is a more comprehensive concept than productivity, taking care in the upper part of its formula of all the possible *outcomes* (not only outputs) produced within the project [BUGL05]. For instance, in CMMI model *productivity* is a ML2 concept (needed for planning a project), while *performance* is a ML4 concept (typically used in more mature organizations where measurement is more widely used).

The goal of this white paper is to discuss, from multiple viewpoints, the pros & cons in using the definition and application of productivity within software projects as currently done, in proposing an alternative view on it, at the aim to provide more accurate estimations and scheduling into new projects, merging a project management-based view with the FSM-based one.

2.2 What is 'project size'?

As said before, there is a general misuse and misunderstanding about which entity is a FSMM is referred to, that is the *product*, not the *project*. Of course, in an ICT project the software solution can represent the main output to be produced, but its size cannot be the same of the project producing such product. Again, another simple evidence besides in the counting entities each FSMM takes into account: the inputs, outputs, queries and files from the solely functional requirements, not all the requirements pertaining to the whole project.

Looking to the more general project management domain, neither the PMI Project Management Body of Knowledge (PMBOK) provides a definition in its glossary or – indirectly – in the text of what can be the size of a project, even if a project is clearly defined. As yet did with products (i.e. ISO 9126 or the same FSM methods compliant to ISO 14143), a unique answer cannot be done, but it could be declined which project *attribute* is to be sized in order to find – from a measurement viewpoint – the proper unit of measure.

Coming back to the Software-Systems Engineering domain, taking a look to the current ISO recognized FSM methods, only the NESMA CPM [ISO05a] uses the term '*project size*' (even if this CPM has no a formal glossary). Reading with attention the NESMA manual it is possible to note that what NESMA calls *project size* is an improper term. Some evidences:

***Where* in the NESMA CPM**	***What***
• Section 2.2, Figure 1	the "project" is wider than the application to be counted with FPA
• Section 3.2	the definition of "project FP count" speaks only of "functionalities", not about the deployment and sizing of non-functional requirements (NFR)
• Indirect evidence	being the ISO/IEC IS 24570:2005 an ISO standard under the WG12, it must be aligned and respectful of ISO/IEC 14143-1:1998(R2007) about the kind of requirements (only the functional ones) to be considered for calculating FP, not all the requirements in a project
• Indirect evidence	Section 4.4: the title of the section is "FPA during a project". If the project would be the entity to be measured, it wouldn't be possible to measure FP "within" the project itself (as Russian boxes)
• Section 4.6, Table 1	what is labelled "project FP count" consider the "living count" for a project, but always related to its functional side, not more than this
• Section 4.6.1	in the scope of "project FP count" it is clearly stated that "it may include one or more applications", intended as sub-projects (see Section 4.6.2)
• Section 4.6.2	in the project FP count procedure, step1 is about the FP count for each sub-system and in step4 it is said that "The size of the project is the sum of the number of FP recorded as a result of steps 1, 2 and 3".

Therefore, this label exists but it is incorrect and in contradiction with the underlying FSM principles, since a FSU is a *product* and not a *project* measure. As in Figure 1, it must be clear the relationship and differences between the container (the glass, in our case the *project*) and its content (the wine, in our case the *product*) and their size by the proper unit of measure.

Fig. 1 – *Container and Content*

Looking at the glossaries from ISO and IEEE standards on Software & Systems Engineering [SEVO08] as well as PMI and other project management associations, right now it does not exist a formal definition for the term "*project size*", neither in PMBOK, where it is used few times within the text, but never defined. The terms useful as inputs to create a new, specific definition are:

- **Software Project** (IEEE Std-1058-1998): *the set of work activities, both technical and managerial, required to satisfy the terms and conditions of a project agreement.*
- **Functional Size** (ISO/IEC 14143-1:2007): *size of the software derived by quantifying the functional user requirements.*
- **User Requirement** (ISO/IEC 14143-1:2007): *description of the set of user needs for the software. Note: user requirements comprise two subsets: functional user requirements and non-functional requirements.*

Therefore a possible new definition, considering the note to the above UR definition, can be[1]:

> **Project Size**: *the size of a software project, derived by quantifying the (implicit/explicit) user requirements referable to the scope of the project itself*

2.3 Structure of this document

This is the logical path proposed: Section 3 discusses the criteria for classifying requirements (only functional requirements are the input for a FSMM), according to ISO 14143-1, and a proposal for refining such taxonomy. Section 4 propose the following step, the logical steps from requirements till the practical tasks in a WBS scheduled and to be performed in a project, providing management criteria for counting separately at least the functional and non-functional effort in a project, because impacting on the effective productivity value and therefore the following schedule and cost budgeting. Section 5 shows by numbers those possible problems using a product functional size unit as the *project* size unit. Section 6 proposes three different alternatives, with pros & cons, for calculating productivity.

[1] Term to be proposed for inclusion in a next revision of the ISBSG Glossary of Terms [ISBS06].

3 Analyzing & Classifying Project Requirements

It is fundamental to define the boundary of this discussion on productivity. The boundary is the *project*, defined as "*a temporary endeavour undertaken to create a unique product, service, or result*" [PMI04]. The initial inputs to a project are its requirements: to properly address them by the right people in a team (e.g. to be analysed, deployed, tested and released to the customer) it is fundamental to understand their nature. In the following sections, some possible classifications are presented.

3.1 ISO/IEC 14143-1:1998 (R2007)

This ISO standard is the first one in the 14143 series, documenting common principles for ISO functional size measurement methods (IFPUG FPA [ISO03a], COSMIC-FFP [ISO03b][2], NESMA FPA [ISO05a] and UKSMA Mark-II [ISO02]). Part 1 states that requirements can be classified into three types - see Figure 2[3] [ISO98]:

- **Functional** User Requirements (**FUR**): "*a sub-set of the user requirements. The Functional User Requirements represent the user practices and procedures that the software must perform to fulfil the users' needs. They exclude Quality Requirements and any Technical Requirements*"
- **Quality** Requirements: "*any requirements relating to software quality as defined in ISO 9126:1991*"
- **Technical** Requirements: "*requirements relating to the technology and environment, for the development, maintenance, support and execution of the software*"

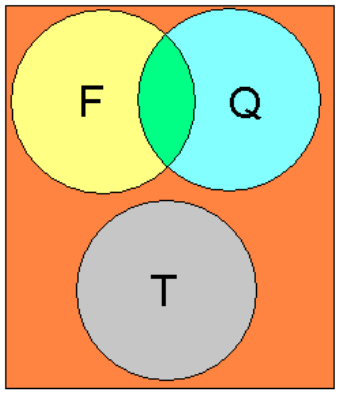

Fig. 2 – *ISO 14143-1 taxonomy for Requirements: Functional, Quality, and Technical (F/Q/T)*

Similarly in IFPUG CPM v4.2, part 2 [IFPU04a]: "*The basis behind "A Framework for Functional Sizing"*[4] *is that there may be multiple sizing methods for different purposes. The functional size could be measured using the IFPUG functional size measurement method for function point analysis, based on the functional user requirements. Other sizing measures can be used to size, for instance, technical requirements. Both result in different size-measures*

[2] The new version for the COSMIC method is v3.0 (now losing the "FFP" in the title) [COSM07].

[3] The overlapping between F and Q circles is due to the presence of the "Functionality" characteristic also in the ISO/IEC 9126-1 standard [ISO01], measuring the software product functionalities in Parts 2-3 from a non-functional viewpoint.

[4] "*A Framework for Functional Sizing*" [IFPU03] is an IFPUG white paper dated September 2003, included and referenced in the IFPUG CPM v4.2.

representing different dimensions of software size: IFPUG-FP for functional size and any other for technical size. While these sizes cannot be added together because they represent different dimensions (like volume and temperature of a room), they can both be used in estimating the effort towards the development of an application or a system".

In the current, updated version of ISO/IEC 14143-1 standard [ISO07], the requirement classification was simplified and the Q/T types were grouped in a unique category, called "non-functional requirements".

3.2 ISO 9000:2000 (R2005)

The ISO 14143-1 formulation takes into account explicit requirements, from a Software/Systems Engineering viewpoint (JTC1/SC7/WG12).

Another ISO view on requirements comes from a management viewpoint (TC 176). The ISO 9000 glossary defines **quality** as the "*degree to which a set of inherent characteristics fulfils requirements*", where "*"Inherent", as opposed to "assigned", means existing in something, especially as a permanent characteristic*".

In CMMI terms, the related practice would be the SP1.1 practice within the Requirement Development (RD) process area (PA), a Maturity Level (ML) 3 PA.

3.3 F/Q/T/O: a possible refinement

Another classification could be between explicit and implicit requirements. For instance, implicit requirement could be what is needed in terms of project planning, project monitoring & control, configuration management, documentation, etc. Therefore a series of organizational and support processes (see §4.2) to be run within a project are not necessarily documented not in an explicit manner by a customer.

Also these requirements will be part of the project scope: therefore the effort required by such processes and activities should contribute in calculating the productivity levels. From Figure 2, the orange area could represent what is required by organizational & support processes activities within a project - see Figure 3.

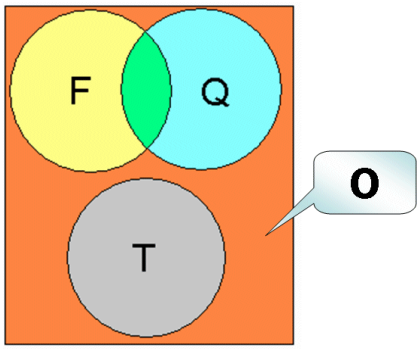

Fig. 3 –*Taxonomy for Requirements: Functional, Quality, Technical and Other (Org + Sup)*

3.4 F/NF: a simpler, basic refinement

Since the refinements above proposed could be not so intuitive, easy to be realized or simply, time-consuming, the minimum level to reach is the separation between the "F" requirements and all the other ones (Q/T/O) that could be aggregated in a generic, unique category, to be referred as "non-functional" (NF), following the ISO/IEC 14143-1:2007 indication.

A high-level rule for classifying the nature of a requirement could be this one: the "what" refers to a "F" requirement, the "how" refers to a "NF" requirement.

If there is a doubt about a unique and not ambiguous nature of a requirement, probably it comprehends more possible sub-requirements and needs to be split in two or more sub-requirements, each one to be classified as "F/NF".

3.5 Why F/NF requirements should be separately managed: the case for Adjustment Factors & Cost Drivers

Coming back to roots, there is some "implicit treatment" of non-functional issues within Albrecht's FPA and other 1st-generation FSM methods. On the one hand, FPA claim to capture only the functional size of a software product, no matter the technology and the way the software is produced. But, on the other hand, the introduction of the Value Adjustment Factor (VAF) and its General Systems Characteristics (GSC) demonstrates the opposite because including them in the calculation means to include a non-functional contribution into a functional size measure. It is sufficient to browse the GSC in order to appreciate the non-functional nature of such drivers (i.e. GSC#6: Online Data Entry, GSC#7 – End User efficiency; …). Another different (but similar) example is given by a well-known estimation model as COCOMO [BOEH81][BOEH00], which scale & cost drivers (i.e. TEAM, PVOL, STOR, AEXP , etc.) represent the expression of non-functional requirements to be reduced to a value expressing a functional view.

In terms of calculation this is easily verifiable:

- In FPA, VAF was used as an adjustment factor with a ± 35% variability from UFP value, with a range between 0.65 and 1.35;
- In COCOMO, the main (functional) value is given by the number of KSLOC (eventually backfired to Function Points), adjusted by the so-called 'cost drivers' (non-functional). The product of all these effort multipliers results in the EAF (Effort Adjustment Effort), where a typical range can be between 0.9 and 1.4.

In both cases, the usage of VAF/EAF (non-functional) as a percentage of UFP/KSLOC (functional) means that non-functional tasks (and effort) were considered less relevant in the project, while they express simply processes and tasks of a different nature.

Paradoxically, if VAF is below the average value (i.e. VAF=0.9), even if the project expressed a certain amount of work effort for those activities related to the 14 GSC, the final (functional) size is lower. Thus, working more, the final size is smaller than the unadjusted one.

Two main consequences:

- From a *technical* viewpoint, the impact is to derived an underestimated *project* size and therefore to make worst estimations if – at least – it is not known the proportion between the functional vs the non-functional size and effort in the project
- From an *economic* viewpoint, the impact is to recognize – with the same number of m/d - lower tariffs for non-functional activities to a provider.

A further evidence about the opportunity and technical correctness of this argumentation can be derived by the rationale of the ISO decision for IFPUG FPA standardization (ISO/IEC 20926:2003) excluding VAF [ISO03a]. Some recent papers stressing the opportunity to do not apply a 'one-size-fits-all' approach are [DEKK08] [LANZ08].

3.6 Non-Functional Requirements: some standards and initiatives

Looking at the technical literature and to the main organizations for standardizations, few information, research and studies have been devoted right now to the analysis and quantification of non-functional requirements [GLIN05]. Anyway, Software & Systems Engineering practitioners are demonstrating a growing interest on this issue, here in the following some standards and on-going initiatives about non-functional requirements:

- **FURPS(+)**: FURPS is the acronym for a software product quality taxonomy – as well as ISO 9126 - by Grady & Caswell [GRAD87] and refined with more attributes into FURPS+ [EELE05]. FURPS stands for **Functionality** (to be split into: Feature Set, Capabilities, Generality, Security), **Usability** (Human Factors, Aesthetics, Consistency, Documentation), **Reliability** (Frequency/severity of failure, Recoverability, Predictability, Accuracy, Mean time to failure), **Performance** (Speed, Efficiency, Resource consumption, Throughput, Response time), **Supportability** (Testability, Extensibility, Adaptability, Maintainability, Compatibility, Configurability, Serviceability, Installability, Localizability, Portability). The "+" addition, is an aid for remembering concerns such as: Design requirements, Implementation requirements, Interface requirements and Physical requirements.
- **ECSS standards**: ECSS (European Cooperation for Space Standardization) is an initiative established to develop a coherent, single set of user-friendly standards for use in all European space activities. Among the several standards produced, 'technical requirements' are diffusely treated. For instance in **ECSS-E-10A,** [ECSS05], become also the **ISO 21351:2005** standard [ISO05c], Sections 6 and 8 discuss about the description and requirements and recommendations for their characterization.
- **IFPUG initiative on "Technical Sizing"**: also IFPUG, by the IT Performance Committee (ITPC), has recently launched a project that will attempt to solve the issue of size estimating of project deliverables, other than Functional Requirements. The project has the aim of defining a framework for technical size. The current issue with Functional Size is that it has not been suitable for sizing the technical requirements associated with a software development project. The technical size will be an addendum to the existing Functional size measure as defined by IFPUG. The goal for this initiative is to define a framework that can be agreed to and supported by both the IFPUG Board as well as IFPUG Members. Preliminary results of this project will be presented to the IFPUG ISMA Conference to be held in September 2008. The project will then create a White Paper that will identify the framework for the technical sizing approach to be approved by the IFPUG Board of Directors and the IFPUG membership.

4 From Requirements to WBS: a process-oriented view

4.1 Requirement-Process-Task

Once elicited and agreed, high-level requirements with the Customer, from a project management viewpoint there is a quite direct translation from requirements to tasks into a Work Breakdown Structure (WBS), to list the activities to be estimated and assigned to the project team members.

Additional information that would be helpful in the planning phase is to know from which process a task is derived and therefore how the effort types by nature are distributed in a certain project, also for estimating & benchmarking purposes [BUGL08b].

The "chain" would be:

HLR → RHLR → (*process*) → task

where *HLR* is the acronym for High-Level Requirements and *RHLR* stands for Refined HLR.

From an operational viewpoint, observing a WBS it is possible to go back in the chain, from tasks to requirements, passing for the processes they are related with. For instance, a task related to a project meeting will be linked – in the CMMI schema – to the PMC (Project Monitoring & Control) process area; a task related to requirement elicitation will be linked to the RD (Requirement Development) process area; and so on.

4.2 Process grouping and effort classification

According to the process model chosen, it is possible to have a different number of process groups. For instance, CMMI has four process groups (Project, Process, Support, Engineering), while ISO/IEC 15504-5 (based on ISO 12207) has five groups (Customer, Engineering, Management, Organizational, Support).

From the ISO 14143-1 classification presented above, it is possible to associate to certain process group a functional or a non-functional nature, as well as to their related tasks.

A *rule of thumb* can be used to classify as functional effort the totality of tasks derived from Engineering processes (both in the CMMI and ISO/IEC 15504 schemas), excluding tasks related to quality and technical requirements, as defined by ISO 14143-1.

Another tip to classify the nature of a requirement can be derived from the role in a project team devoted to deploy such requirement: a database administrator (DBA) typically will execute tasks related to a "NF" requirement, while an analyst/programmer tasks related to a "F" requirement, and so on.

4.3 Deriving project effort by types

According to the above mentioned rule, it can be possible to obtain from each WBS an approximation of the amount of effort planned and actually spent by type at different levels of grouping:

- A simpler one (Functional vs Non-Functional effort);
- A more refined one, splitting the NF part into more sub-groups (F/Q/T or F/Q/T/O).

Whether it is difficult to assign a well-identified nature to a task, probably it will be possible to split such task in two or more ones, each one with a clearer and well-identified nature. For instance, a generic "Project management" task could be split into two sub-tasks (Planning and Monitoring & Control); similarly a generic "Testing" task can be refined into a testing level (unit, integration, system), by type (black, grey, white), and so on.

Furthermore, as PMBOK and best practices on PM suggest, tasks should be not too long, in order to be more manageable and verifiable, providing the possibility to take quickly corrective action when required. The (approximated) distribution of the project effort by type can improve the planning and monitoring & control of a project, as discussed in detail in [DERY05].

A further deployment of such distributions is described in [BUGL08b], treating about project *effort proportions*.

5 Productivity calculations

5.1 Productivity calculation for software projects

Moving from the generic definition of productivity (*output /effort needed to produce such output*) valid for any domain, the instantiation within the Software Engineering domain in the '70s considered as the main measurable item the number of LOC: productivity was then calculated as the ratio LOC/project effort. The "evolution" of such concept during the late '70s and '80s with the introduction of Function Point Analysis (FPA) led to the calculation of productivity as the ratio between Functional Size /project effort[5].

.The IFPUG GSM (Guidelines to Software Measurement) [IFPU04b], provides several ratios based on the number of Function Points, in the so-called **PDR** (Productivity Delivery Rate), as in the ISBSG benchmarking data [ISBSG07]. It must be noted that PDR presents the inverse formula than Productivity as above defined, measuring therefore the effort needed to produce a functional size unit.

5.2 An example

Objective: calculate the productivity level for a web project at a certain time for monitoring the project and eventually take some actions, based on the following data:

- Functional (*product*) size: 980 Function Points (FP)
- (*project*) effort: 1200 man/days (m/d), supposing to subdivide the overall value, according to ISO 14143-1 requirements taxonomy, in three parts (last column is calculated in order to provide an approximated balancing among parts):

Req. Type	Effort (m/hrs)	Effort (m/d)	Effort (%)
F – Functional	5600	700	58.3
Q – Quality	1600	200	16.7
T – Technical	2400	300	25.0
Total	**9600**	**1200**	**100.0**

- Productivity: 980 FP / 1200 m/d = **0.82 FP / m/d = 0.102 FP / man-hour**

In comparison to available external benchmarking data [ISBSG07][DCG07] for web projects with an average productivity in the range of 0.125 to 0.1625 FP per hour (or from 1.0 to 1.3 FP per man-day), this project seemed to have a productivity level lower than average and some corrective actions should be planned.

Looking at these benchmarks, a current limitation is due to the data gathering process within ICT organizations, which does not take into account such effort granularity. Thus it is not possible to know which the effort distribution by type is and which impacts from the technical and economical viewpoints should be taken. For instance, how could a project manager determine if the current project falls in the same hypothesis of the projects considered to derive the benchmark data? If the project effort is gathered in a unique group with no distinctions per type, are we comparing apples with oranges or apples with apples? This is the research question we are going to analyze, searching for answers.

[5] The well-know *productivity paradox* formulated by Jones [JONE97] provided the evidence of considering a functional size measurement unit, solving the two paradoxes about productivity and costs for a software project, stressing that must be a proportionality between what produced by a provider and what paid by a customer.

5.3 Entities to be measured

Figure 4a shows the common classification of measurable entities (IPO – Input / Processing / Output) [BUSH90] and Figure 4b, the STAR classification including two more upper-level entities (project; organization) [BUGL02].

Analysing the productivity formula as defined above, it can be observed that the upper and lower parts of such ratio refer to different entities: the upper part of the productivity ratio is a (software) *product* measure and the lower part is a *project* one.

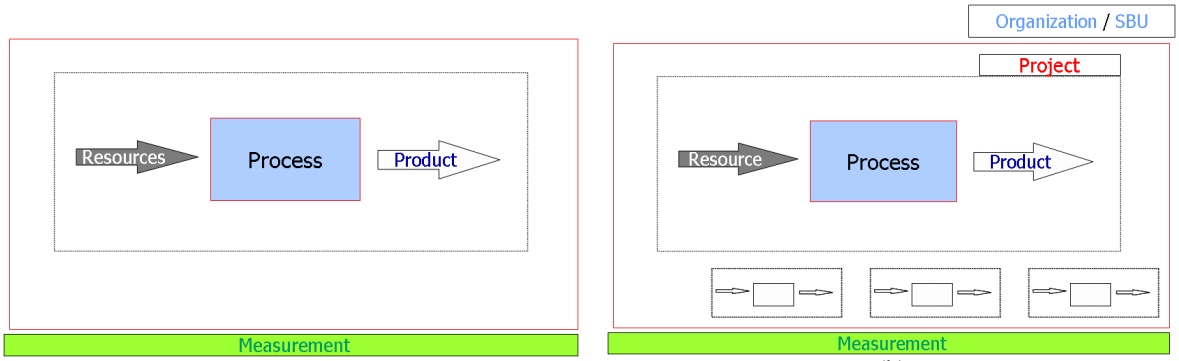

Fig. 4 –*IPO (a) and STAR (b) taxonomies*

In fact, a FSMM unit represents the functional size of a software *product*. And a software product is – undoubtedly – the main output of a software project, but not necessarily the solely one. It is sufficient to look at a SPI model such as CMMI or ISO/IEC 15504 (aka SPICE) in the list of possible work products expected to be managed in each Process Area. For instance, user manuals are other possible project' deliverables, but they are not measurable with Function Points or another FSMM. So, a project with a larger amount of non-functional requirements to be satisfied would not produce more functional units but more effort spent for performing the needed activities. The practical effect would be the determination of a lower productivity value (as currently computed), even if it is not necessarily true, because we would be simply doing other tasks (that is the case of the so-called "*Zero-FP projects*" [DEKK07]).

So next question is: for which entity are we committed to measure and estimate? Have we to measure the capability to produce the main final product or the overall project by a more comprehensive viewpoint than the solely functional one? In fact we are in charge to measure is the *project*, also by their work products, but taking care to define its *scope* (and boundary) as the first action (see PMBOK, CMMI PP SP1.1, etc.).

Thus this raises a few questions more:

- *Is the productivity ratio – as currently calculated - meaningful or not from a management perspective?* A first thought on the current definition of productivity is that it could be underestimated, because it puts in relation only some outputs against the overall effort is based under the assumption the such produced quantity is representative of the whole project. The more the non-functional effort (that one not referable to a FSM unit), the lesser the productivity level, even if not strictly related. Figure 5 tries to put in evidence in a graphical way such concept.

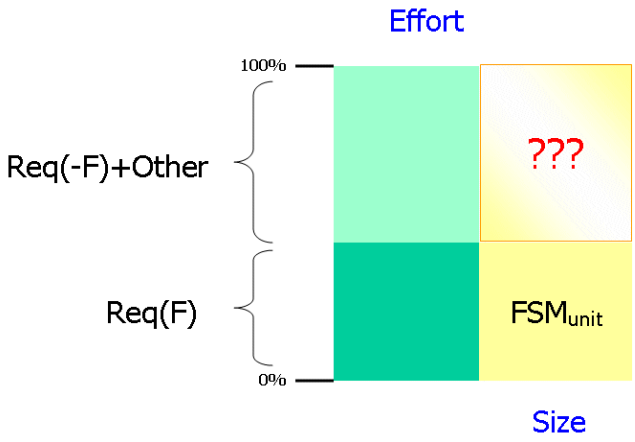

Fig. 5 – *Effort & Size: possible relationships*

The predominance of the functional side in a software project from the user's viewpoint produced the wrong perception that a functional size unit (and before the number of LOC) represented the project. Having gained during the years a more mature knowledge and awareness on Project Management and Software-Systems Engineering, this interpretation could risk to do not represent anyway a proper picture of what a software project express right now.

- *What about the technical impacts it could have for a project manager*? Suppose that the productivity calculated in the above described way. Imagine that the Customer is asking to write a user manual (not a software help-on-line) or to run new stress tests on the systems. All these types of activities are classifiable as non-functional requirements, according to ISO 14143-1. Using this productivity definition, the more the non-functional effort (that one not referable to a FSM unit), the lesser the productivity level, even if not strictly related. So, it is fundamental to gather and properly understand the distribution of effort by nature, at least the proportions between functional vs. the non-functional efforts. The *technical impact* could be to wrongly schedule the project, staffing more people that needed or asking a wrong number of people by skill types: a SOA architect will run a non-functional analysis and such effort should not be absorbed in the calculation of the productivity level, differently from the case of an Analyst/Programmer. Again, the effort for a Usability Engineer will typically address – according to ISO 14143-1 taxonomy - a Quality requirement, therefore also such effort should not be related in terms of effort to a functional (product) size.

- *What about the business impacts it could have for a project manager*? If a project manager does not know the approximated distribution between the project functional and non-functional effort and it is adopted – as in many contracts – the cost/FP as the main (or the solely) economic measure to evaluate the price for a software project between the customer and the provider, the *business impact* could be to do not recognized any economic value for non-functional activities run, or a value lower than expected when VAF-like elements were introduced.

6 A different view on Productivity

In order to overcome some of the technical and business limitations in the current way of calculation productivity, the following three alternatives have been identified: the first two use *product* measures in the upper part of the productivity formula, and the third one uses a *project* measure.

6.1 Proposal #1

Using the current interpretation *("functional product size unit"/ effort*), but gathering at least the project functional vs. non-functional effort (in absolute value and %), in order to undertake the proper actions during the project lifetime. In such way, it will be possible to identify a cluster of similar projects from a historical database also filtering by such values and avoiding low-quality dataset for *estimation* purposes. Again, it will be possible to refine the indication about the number of FTE (Full Time Equivalent) needed in the project in two macro-groups for *staffing* purposes. Applying the project data from the previous example presented in Section 4.1:

- Functional (*product*) size: 980 Function Points (FP)
- (*project*) effort: 1200 man/days (m/d) subdivided – according to ISO 14143-1 requirements taxonomy in:

Req. Type	Effort (m/d)	Effort (%)
F – Functional	700	58.3
Q – Quality	200	16.7
T – Technical	300	25.0
Total	**1200**	**100.0**

the following information would be obtained:

- Productivity: 980 FP / 1200 m/d = **0.82 FP / m/d**
- **Functional** productivity: 980 FP / 700 m/d = **1.40 FP / m/d**

Having this double information, in this case it is possible to schedule in a more realistic manner the project, taking care of the distribution of effort types (c.a. 60% F; 40% NF), with a functional productivity quite higher than the nominal one. The suggestion would be to include the proper amount of human resources in the project team according to the split of effort and to schedule in a shorter period the deployment of the functional side of the project, due to the higher functional productivity.

6.2 Proposal #2

Using the current interpretation *("product" size units / effort*), but calculating different productivities according to the defined types of requirements (i.e. F/Q/T/O), each one as the ratio between a proper size measure and only the related effort, therefore a functional productivity, as the ratio between FSM unit and the project functional effort; a quality productivity as the ratio between one (or more) quality measure(s); and so on. Applying the project data from the previous example presented in Section 4.1 and supposing there would be for Q/T parts equivalent unit of measures than FP for the F part, conventionally called here Quality Points (e.g. it could be an

elaboration from the ISO 9126 standards) and Technical Points (in this case it will be the final result for the IFPUG work on the "Technical Size", as discussed in Section 3.6)[6]:

- Functional (*product*) size: 980 Function Points (FP)
- Quality (*product*) size: 150 Quality Points (QP)
- Technical (*product*) size: 195 Technical Points (TP)
- (*project*) effort: 1200 man/days (m/d) subdivided – according to ISO 14143-1 requirements taxonomy in:

Req. Type	Effort (m/d)	Effort (%)
F – Functional	700	58.3
Q – Quality	200	16.7
T – Technical	300	25.0
Total	**1200**	**100.0**

the following information would be obtained:

- Productivity: 980 FP / 1200 m/d = **0.82 FP / m/d**
- **Functional** productivity: 980 FP / 700 m/d = **1.40 FP / m/d**
- **Quality** productivity: 150 QP / 200 m/d = **0.75 QP / m/d**
- **Technical** productivity: 195 TP / 300 m/d = **0.65 TP / m/d**

In such way, it will be possible to refine more in detail the tentative schedule by more tasks family as well as to have more data for a proper staffing of "Q" people (i.e. quality assurance, usability, …), "T" people (architects, database administrators, …) or "O" people (team leaders, translators, documentation writers, …), according to each Organization's list of roles and job titles.

6.3 Proposal #3

Applying the productivity formula comparing a *project size* with the project overall effort. "Project size" is a concept often discussed, but quite often not yet properly defined and measured. Also in PMBOK, this term appeared several times but with no definition, neither in the glossary. A possible technique, respecting the proposal of definition we did in Section 2.2 of this document, is **PSU** (*Project Size Unit*) [BUGL07], a project management-based technique.

Applying the project data from the previous example presented in Section 4.1, with a (project) effort of **1200** man/days (m/d) and supposing a 'project size' equal to 372 PSU[7], the following information would be obtained:

- (**Project**) Productivity: 372 PSU / 1200 m/d = **0.31 PSU / m/d**

This figure can be particularly helpful more in the early stages of a project for planning purposes, when it is not available an information about the effort type split (as presented before), looking only from a project management viewpoint to an overall project effort. But it can be integrated also, during the project lifetime, with a FSMM in software projects.

Using PSU, a further level of detail could be a series of productivity figures by task types (M/Q/T: Management/Quality/Technical), referring to a different classification based on primary and support & organizational processes, as specified in the PSU Measurement Manual.

[6] Those two values are just for completing the numerical examples. Obviously, in order to take the proper corrective actions, the interpretation of such values should be related to some reference (internal-external) benchmarking values.

[7] Assumptions: 364 tasks distributed by complexity as follows (H= 6; M= 8; L= 350), with the following weights (H= 1,8; M= 1,4; L= 1,0). For the calculation rules, please refer to [BUGL07].

7 Technical & Business Impacts

Here in the following a couple of issues on which paying attention, both for the technical and business viewpoints.

7.1 Average & Median values

Looking at statistics, the *average mean* is one of the most used figures. But it can lead to some misinterpretation if not compared to other values for the decision-making process, for instance to the *median*. Suppose to have this short list of projects:

Project Id.	FSM unit (#)	Effort (m/d)	Productivity (FP/effort)
Prj001	560	600	0.93
Prj002	600	640	0.94
Prj003	1000	1200	0.83
Prj004	200	400	0.50
Total	**2360**	**2840**	
Max	1000	1200	0.94
Median	**580**	**620**	**0.88**
Average	**590**	**710**	**0.80**
Min	200	400	0.50

Looking at the average values, if not compared with the median, it would seem that a reference value for such project cluster in terms of productivity could be 0.80FP/m-d. But looking at the whole data series, Project *Prj004* is an outlier, with quite the half of the productivity of the first two projects. With this more value, it would be suggestible to consider a higher productivity level in estimation assumptions because more representative also of the frequencies within the data distribution.

The same discussion could be also posed in terms of costs for the people composing the team and fixing the average price per day of staffed people. But what about the higher costs of a high-skilled non-functional role such as a project manager against lower tariffs for an analyst-programmer? Having the two numbers can help in determining the best choice, more than simply having the solely average mean.

7.2 Cost per unit

Another question related to the entity to be measured is: how much does it cost each *project size unit*? During the years, several contracts used (and currently use) the *Project Cost/FP* as the solely economic measure for managing costs within a software project. But being FP a *product* (not a *project*) size unit, measuring only the functional size of a software solution, it means that in this case the non-functional part of the project – if not measured by other Q/T/O measures – is not paid. Coming back to the previous example:

- Functional (*product*) size: 980 Function Points (FP)
- (*project*) effort: 1200 man/days (m/d) subdivided in:

Req. Type	Effort (m/d)	Effort (%)
F – Functional	700	58.3
Q – Quality	200	16.7
T – Technical	300	25.0
Total	**1200**	**100.0**

- Productivity: 980 FP / 1200 m/d = **0.82 FP / m/d**
- Total Project Cost: 1,000,000 $
- Cost/FP: 200,000 $ / 980 FP = **204 $ / FP**

But translated in terms of effort (m/d), if the functional part of the project represents the 58.3% of the overall project effort, the remaining 41.7% is not included in such tariff[8].

In fact, with such productivity (0.82 FP/m-d), the worked days to be paid would be 800 (980*0.82) and not 1200 m/d, as effectively spent in the project.

Again, there is still the feeling that the non-functional activities would value (and therefore cost) less than the functional ones, while they are simply different in nature and to be treated separately, both from a technical and economical viewpoint.

A possible solution would be to apply – as in the Proposal #3 – a unit of measure sizing the whole project. In this way, the cost issue would be properly represented between two parts simply sharing the same definitions for calculating the project size unit, whatever the technique applied.

[8] Discussed on the IFPUG Bulletin Board: http://www.ifpug.org/webforum/discus/show.cgi?1780/9214. An extended discussion will be proposed on the first 2008 issue of IFPUG's "Metrics Views" newsletter.

8 Conclusions & Prospects

One of the most discussed questions in ICT projects using a FSMM are: which effort is in or out the productivity calculation? Should the time for writing a User Manual be included? And the effort spent for the Architectural Analysis?

But the counter-question to those frequent questions could be simply: are we measuring software or something else? Is the nature of the activity to be run functional or not, according to ISO definitions? Is it meaningful to relate a part of the project to its whole effort?

It does not exist a single truth, but each thing can be seen from several, concurrent viewpoints. In any case there are quantitative *checkpoints* that help to verify if we are achieving or not our estimation goal, as ARE and MRE values for each single project, and so on.

The goal of this white paper is not to catch the truth, but possibly to stimulate a discussion of what should be a consistent application of the productivity definition, considering the effects from different stakeholders' view.

Getting a picture of the project not representative of its underlined status is not useful neither from a technical viewpoint for scheduling and staffing it, nor from an economic viewpoint because estimations will be out of the accepted ranges and consequently from a social viewpoint, increasing the probability of unstable requirements or time spent to re-discuss contract's clauses. A good contract must satisfy both parts, and the reason for success besides a proper and objective definition (the more as possible) of the subject of the contract, possibly with no undesired side-effects.

A wrong (or not proper) application of a FSMM for contractual purposes risk to reduce the trust such methods has gain during these 30 years, while such methodologies should be managed and used for what they can positively bring into the management of a project.

Next step will be to measure the *project*, also by its products, but with an enlarged view. The suggestion is to analyze a project taking care and managing separately its parts according their different natures (the '*divide et impera*' principle), but not loosing the bird's eye view on it, respecting and coordinating the stakeholders' viewpoints.

--- End of the Document ---